Estimation and Planning Mistakes

- **Underestimating complexity, cost, and/or schedule**
  - Use historical data and expert judgment to estimate accurately.
- **Abandoning planning under pressure**
  - Stick to planning to avoid chaotic code-and-fix mode.
- **Overly aggressive schedules**
  - Set realistic schedules based on historical data and project complexity.
- **Wasting time in the "fuzzy front end"**
  - Streamline the approval and budgeting process.

Communication and Stakeholder Engagement Mistakes

- **Poor communication**
  - Hold regular meetings and ensure clear documentation.
- **Not engaging stakeholders**
  - Include stakeholders in planning and review sessions.
- **Insufficient user input**
  - Ensure active involvement of end-users throughout the project.

Project Management Mistakes

- **Lack of oversight/poor project management**
  - Appoint experienced project managers and conduct regular reviews.
- **Adding developers to a late project**
  - Avoid adding developers late in the project to prevent further delays.

Quality and Risk Management Mistakes

- **Poor quality workmanship**
  - Implement quality assurance processes and conduct regular code reviews.
- **No risk management**
  - Identify risks early and develop mitigation plans.
- **Ignoring system performance requirements**
  - Define and monitor performance requirements throughout the project.
- **Poorly planned/managed transitions**
  - Develop detailed transition plans and involve all relevant parties.

Agile Manifesto

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

Agile Principles

- Satisfy the customer with continuous delivery.
- Welcome changing requirements.
- Frequent delivery of working software.
- Daily collaboration between business and developers.
- Build projects around motivated individuals.
- Face-to-face conversation for communication.
- Working software as progress measure.
- Promote sustainable development.
- Continuous attention to technical excellence.
- Simplicity is essential.
- Best architectures emerge from self-organizing teams.
- Regular reflection and adjustment.

Scrum Roles

- Product Owner:
  - Maximizes product value.
  - Develops and communicates Product Goal.
  - Creates and prioritizes Product Backlog.
  - Ensures transparency and understanding of Backlog.
  - One person, not a committee, with leadership role.
- Scrum Master:
  - Facilitates Scrum process, resolves impediments.
  - Creates self-organization environment.
  - Captures empirical data, shields team from distractions.
  - Enforces timeboxes, keeps artifacts visible.
  - Promotes improved practices, has leadership role.
- Development Team:
  - Develops product, self-organizing, cross-functional.
  - No titles, no sub-teams, no specialized roles.
  - Long-term, full-time membership, 7 ± 2 members.

Scrum Events

- Sprints
- Sprint Planning Meeting
- Daily Scrum Meeting
- Sprint Review Meeting
- Sprint Retrospective

Scrum Artifacts

- **Product Backlog**
  - Prioritized list of features.
  - Updated regularly.
  - Visible to all stakeholders.
  - Owned by Product Owner.
- **Sprint Backlog**
  - List of tasks for current Sprint.
  - Owned by Development Team.
  - Updated daily.
  - Created during Sprint Planning Meeting.
  - Decomposed from Product Backlog.
- **Burndown Charts**
  - Graphical representation of work remaining.
  - Updated daily.
  - Shows progress towards Sprint Goal.
  - Helps identify issues early.
  - Used to forecast project completion.

Accidental vs Essential Complexity

- Essential complexity: - Inherently difficult problems with no known solution.
- Necessary accidental complexity: - Example: project management.
- Unnecessary accidental complexity: - Waste, Lean, MEI (minimum essential information).

Best/Good/Recommended Practices

- "Best Practice": - Consistently improves productivity, cost, schedule, quality, user satisfaction, predictability.
- Best Practices (Glass, 2004): - Development teams repeat mistakes. - Best practice documents regurgitate textbook material. - Growing field's wisdom not increasing.

Agile Sweet Spots

- Dedicated developers.
- Experienced developers.
- Small co-located team.
- Tools for testing and configuration management.
- Easy user access.
- Short increments and frequent delivery.